Meta-modelling for ontology development and knowledge exchange

Mariano Fernández-López
Laboratorio de Inteligencia Artificial
Facultad de Informática
Universidad Politécnica de Madrid
Campus de Montegancedo sn.
Boadilla del Monte, 28660. Madrid, Spain.
Tel: (34-91) {336-66-05, 336-74-39}
Fax: (34-91) 3-52-48-19
Email: mfernandez@fi.upm.es

ABSTRACT

One of the sources of heterogeneity of ontologies is that different ontologies have different necessities of modelling. This paper presents a bi-phase method to deal with these different necessities. Phase I of the method models how to model the ontology, obtaining a meta-model. Such meta-model can be expressed in LBIR, a formal and declarative language that has been specifically designed for this task. To save resources, a reference meta-model that can be modified and reused is provided. During phase II of the method, the ontology is modelled following the meta-model obtained in the first phase. Furthermore, a tool (called ODE) provides software support to the method. Such tool generates SQL schemas from LBIR, and allows the modelling of the ontology following the selected meta-model. This approach eases the interoperability between groups located in different geographical locations that have to build the same ontology, since the meta-model to be used can be exchanged through LBIR.

KEYWORDS

Ontology, meta-model, modelling, method, LBIR, ODE.

1. EXPOSITION OF THE PROBLEM

Even though Ontological Engineering is a very young area in Artificial Intelligence, there exist some methodological proposals for building ontologies: Uschold and King's methodology [Usc95], Grüninger and Fox's methodology [Grü95], METHONTOLOGY [FeG99], etc. A study and analysis of methodologies for building ontologies can be found at [Fer99]. This study shows that METHONTOLOGY is currently the most mature methodology.

Presently, methodologies do not propose to adapt the mechanism of modelling to the different ontologies to be built. However, our experience in different projects (the (KA)^2 initiative [Ben98], the multidisciplinary project AM9819 about environmental pollutants, etc.) show that different domains should be modelled in different ways. Table 1 shows the components that have been used in different ontologies. We can see that there are variations from some ontologies to others. Some ontologies have been built using a lot of attributes and no relations, others have been built using constants, some of them have first order logic formulas, but others do not, etc.

Apparently, one solution to this problem would be to consider all the "necessary" components (concepts, attributes, first order logic formulas, constants, etc.) when an ontology had to be built. Nevertheless, such solution has the following drawbacks: (1) Our experience has shown it is possible that need for a component is not perceived a priori, that is, it is possible the necessity of a component is only detected when it is needed in an ontology. (2) New research about modelling can provide new components and new ideas about how to use old components. (3) Considering non-useful components when an ontology is built can cause confusion in modellers, and especially when they are not very experienced.

Besides flexibility in the components to be used during the modelling, the knowledge should be presented in different ways to different experts.

Summarising, **a rigid way to model brings us back to the classic knowledge-acquisition bottleneck** [Eri99].

Ontology	**Domain**	**Concepts**	**Instance attributes**	**Relations**	**Constants**	**First order logic formulas**	**Arithmetic formulas**	**Instances**	**TOTAL TERMS**
CHEMICALS.1	Chemical	10	6	0	0	0	1	20	37
CHEMICALS.2	Chemical	16	22	0	0	27	3	103	173
CHEMICALS.3	Chemical	16	20	0	2	27	1	103	169
$(KA)^2$ restructured	Knowledge acquisition community	78	12	47	0	0	0	102	239
Reference Ontology	Ontologies	23	70	9	0	0	0	8	110
Standard Units restructured	Measure units	22	3	0	2	0	1	65	93
Monatomic ions	Environmental ions	62	11	3	0	6	0	0	82
Silicates	Silicates	84	17	8	0	0	0	0	109
Hardware	Laboratory of Artificial Intelligence's hardware	49	56	0	0	0	0	56	190
ELLOS Ontology	Catalogue of clothes	8	16	6	0	0	0	20	48
Tradezone Ontology	Catalogue of products in general	9	3	3	0	4	0	0	22
SNCF Ontology	Travels and hotels	13	37	4	4	2	2	1	69
FIDAL Ontology	Contracts	6	15	8	0	1	0	7	37

Table 1. Components used in the ontologies developed with the bi-phase method

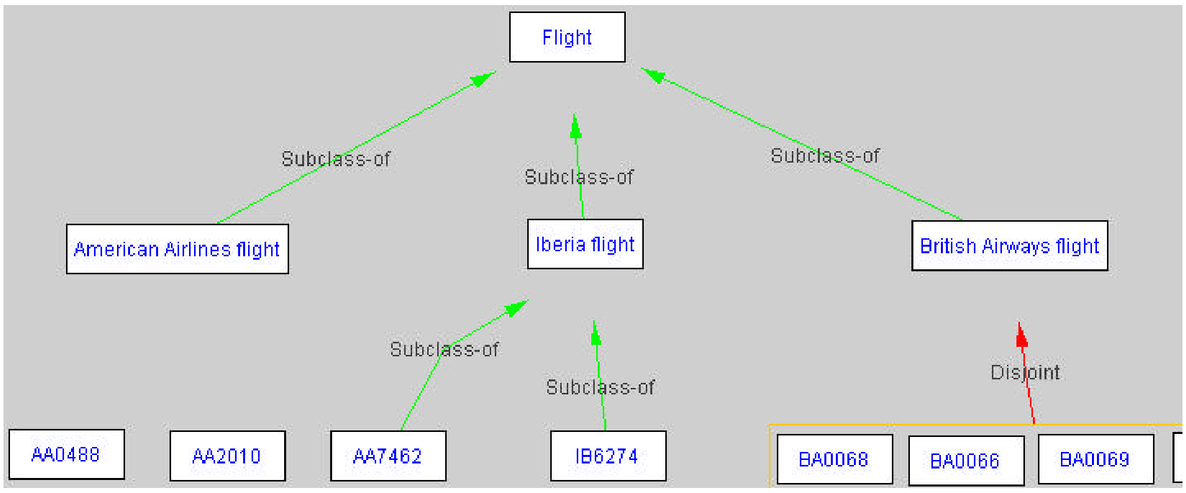
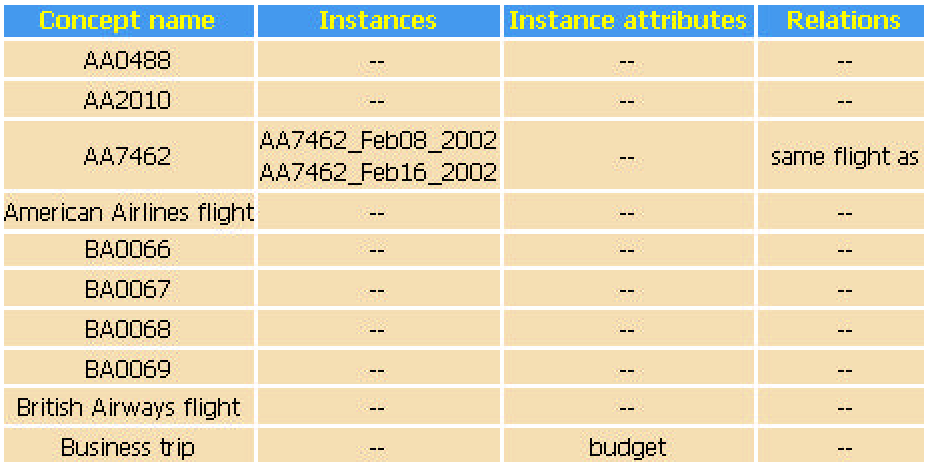


Figure 1. `Concept classification tree` *in the domain of flights*



Concept name	Instances	Instance attributes	Relations
AA0488	--	--	--
AA2010	--	--	--
AA7462	AA7462_Feb08_2002 AA7462_Feb16_2002	--	same flight as
American Airlines flight	--	--	--
BA0066	--	--	--
BA0067	--	--	--
BA0068	--	--	--
BA0069	--	--	--
British Airways flight	--	--	--
Business trip	--	budget	--

Table 2. `Concept dictionary` *in the domain of flights*

Focussing on the case of METHONTOLOGY, it proposes to carry out the following steps to develop an ontology: *specification* in natural language, *conceptualisation* using tables and graphs, *formalisation* (e.g. using frames), and *implementation* (e.g. using the Ontolingua language [Far97]). According to the METHONTOLOGY viewpoint, conceptualisation is the modelling at the knowledge level [New82], hence, the knowledge is modelled independently of the implementation language to be used[1]. The proposed tables and graphs allow modelling concepts, attributes, first order logic formulas, etc., and they are thought to be manipulated by experts in the domains to be modelled. Figure 1 presents an example of a graph: a `concept classification tree`, and table 2 is an example of `concept dictionary`. Tables and graphs in METHONTOLOGY are not fixed, since the engineer can use tables or graphs that can be different to the proposed ones by the methodology. However, METHONTOLOGY does not propose a precise way to specify how the tables and the graphs to be used during the conceptualisation are. Besides, this methodology does not propose how to add a new type of table, how to add a new field to a type of table, how to

[1] Such idea of conceptualisation is inspired in the Hayes-Roth and colleagues' approach [Hay83].

delete one of the types of the proposed graphs, or how to elaborate a completely new modelling way with completely new graphs and tables. Therefore, if several groups in different locations have to build an ontology collaboratively, **there are problems to agree and exchange the characteristics of the tables and graphs to be used** (see figure 2).

In the following sections, a solution to these problems will be presented. Section 2.1 will present the bi-phase method and, section 2.2, its software support: ODE. The paper will finish with the conclusions and future trends.

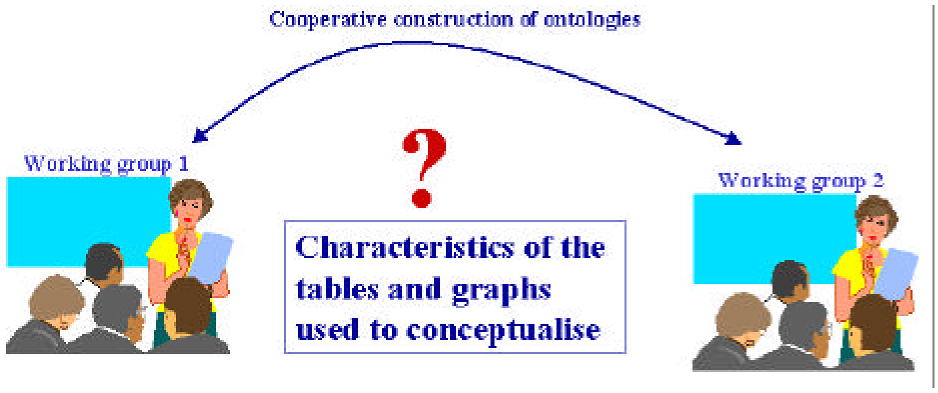

Cooperative construction of ontologies
Working group 1
Working group 2
?
Characteristics of the tables and graphs used to conceptualise

Figure 2. Problems in collaborative construction when the characteristics of tables and graphs are not clearly specified

2. THE PROPOSED SOLUTION

2.1. THE METHODOLOGICAL LEVEL OF THE BI-PHASE SOLUTION

To allow a more flexible modelling of ontologies and to ease the exchange of characteristics of tables and graphs, the **bi-phase method** proposes to model how to model the ontology. Until now, the purpose of the ontology engineer was to model some parts of the world, for example, flights, chemical elements, etc. (see figure 3), however, with the bi-phase method, modelling the process of modelling is also recommended, that is, building a **meta-model** is also proposed. Particularly, the part of the modelling process to model is the conceptualisation, which is the base of the remainder steps of the modelling.

The bi-phase method follows the METHONTOLOGY approach, although in two levels. On the one hand, during phase I, the ontology conceptualisation process is specified in natural language, conceptualised using tables and graphs (called in this phase **meta-tables** and **meta-graphs**), formalised using a formal language, and implemented in SQL (see figure 4). Thus, the result of this first phase is a meta-model presented in meta-tables and meta-graphs, in a formal language, and in SQL. The steps of this phase are called: **meta-specification**, **meta-conceptualisation**, **meta-formalisation** and **meta-implementation**. On the other hand, phase II carries out the specification, conceptualisation (following the meta-model obtained in phase I), formalisation and implementation of the ontology. As you can see, in this bi-phase method, there is a modelling both at Newell's knowledge level and symbolic level during phase I as well as phase II.

To facilitate the building of meta-models, a **reference meta-model** is proposed. It is possible to modify this reference meta-model according to the modelling needs of each ontology. Such meta-model is expressed by means of meta-tables and meta-graphs, and it is also formally expressed. The reference meta-model allows building ontologies with: concepts, class and instance attributes, facets of such attributes, relations, first order logic formulas, arithmetic formulas, constants, and instances. These components appear in the reference meta-model because each one of them have been used in some of the ontologies developed during the experimentation. Besides, we have checked that the reference meta-model contains the static components of the classic languages for ontology development (Ontolingua, OKBC, OCML, FLogic and LOOM). We say static components because we do not consider rules and procedures. This reminds as future work.

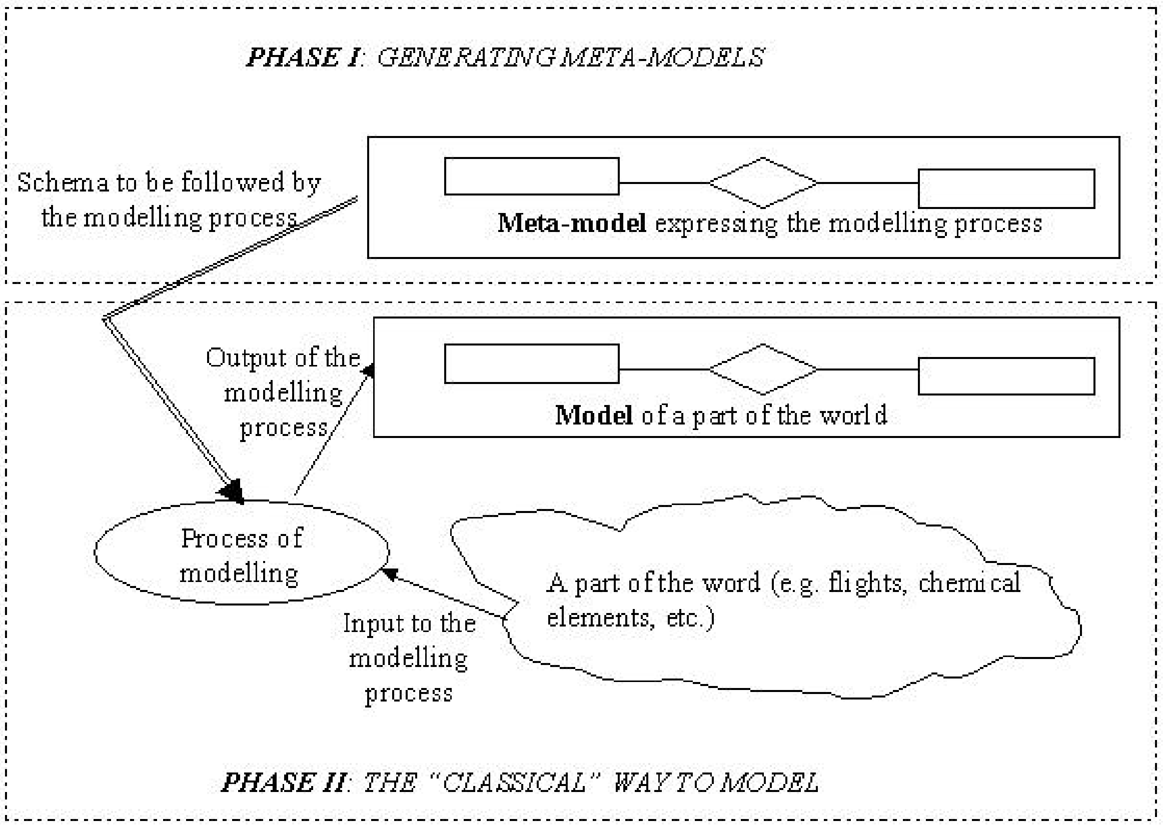

PHASE I: GENERATING META-MODELS
Schema to be followed by the modelling process
Meta-model expressing the modelling process
Output of the modelling process
Model of a part of the world
Process of modelling
A part of the word (e.g. flights, chemical elements, etc.)
Input to the modelling process
PHASE II: THE "CLASSICAL" WAY TO MODEL

Figure 3. General overview of the ontology development using meta-models

We have also developed a tool, called ODE, in order to provide software support to the bi-phase method. Ode is especially designed to facilitate the application of the method. However, it is not the only tool allowing flexible modelling, since Protégé-2000 [Fri00] permits the user to redefine its components (made by classes, slots, etc.).

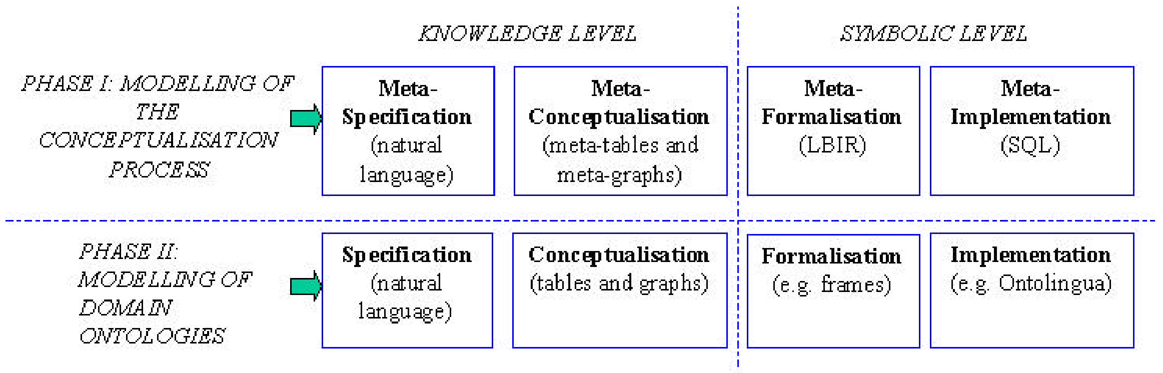

KNOWLEDGE LEVEL
SYMBOLIC LEVEL
PHASE I: MODELLING OF THE CONCEPTUALISATION PROCESS
Meta-Specification (natural language)
Meta-Conceptualisation (meta-tables and meta-graphs)
Meta-Formalisation (LBIR)
Meta-Implementation (SQL)
PHASE II: MODELLING OF DOMAIN ONTOLOGIES
Specification (natural language)
Conceptualisation (tables and graphs)
Formalisation (e.g. frames)
Implementation (e.g. Ontolingua)

Figure 4. Bi-phase method to build ontologies

In [Fer01] a complete description of the method is presented. Such description includes the tasks to be performed, the inputs, the outputs and the participants. This description includes a way to manage the changes in meta-models, even when an ontology is being developed with such meta-model and new necessities are detected. There is also a description of the architecture of ODE. The method and the tool have been tested in the above mentioned projects (the (KA)^2 initiative, the multidisciplinary project AM9819 about environmental pollutants, etc.). 10 different meta-models have been built with a total of 33 additions, removals and modifications with regards the reference meta-model; such meta-models have been used in 11 different domains: chemical elements (169 terms with 27 first order formulas), knowledge acquisition community (239 terms with no first order formulas), hardware (190 terms with no first order formulas), ontologies (110 terms with no first order formulas), measure units (93 terms with no first order formulas), monatomic ions (82 terms with 6 first order formulas), silicates (109 terms with no first order formulas), catalogues of cloths (48 terms with no first order formulas), travels (22 terms with no first order formulas), hotels (69 terms with 2 first order formulas) and contracts (37 terms with 1 first order formula). Other meta-models have been built containing meta-graphs and meta-tables to model databases, other meta-models contain meta-graphs to model tasks, and other meta-models even contain schemas of bills, invoices, etc. as meta-tables.

In the following sub-sections, a brief description of the steps of phase I will be presented.

2.1.1. Meta-specification of the conceptualisation process

During phase I, the meta-specification describes, in natural language: (a) what *tables and graphs* will be used during the conceptualisation of the ontology; (b) the *recommended order* to fill in the tables and to build the graphs; and (c) the *consistency verification rules* between tables, between graphs, and between tables and graphs. For example, it can be (meta-)specified that a graph to be used during the conceptualisation is the `concept classification tree`, that the nodes of this graph are `concepts`, and that the edges are `subclass of`, `subclass in a disjoint partition`[2], and `subclass in an exhaustive partition`[3]. Besides, it can be also specified that a table to use during the conceptualisation of the ontology is the `concept dictionary`. The possible fields of such table would be: `concept name`, `instances`, `instance attributes`, etc. Concerning the recommended order, it should be said that the elaboration of the concept classification tree should begin before starting the concept dictionary. And with regard to the consistency verification rules between the concept classification tree and the concept dictionary, all the concepts of the tree should be in the concept dictionary and vice versa.

2.1.2. Meta-conceptualisation of the conceptualisation process

For (meta-)conceptualising in phase I, the bi-phase method proposes: (a) a set of meta-tables to describe the tables and graphs to be used during the conceptualisation in phase II; (b) a meta-graph to describe the order in the conceptualisation in phase II; (c) and meta-tables and meta-graphs to describe the consistency verification rules. Thus, for example, the `meta-tables of node description`, and the `meta-tables of edge description` are proposed to define the details of the graphs, and the `meta-tables of field description` are proposed to define the details of the tables. For instance, meta-tables 1, 2 and 3 show the description of the taxonomy and of the concept dictionary, used both in the examples of section 1.1.1. In all these meta-tables, the meta-field `symbol` is filled in with abbreviations. In the case of meta-table 2, which describes a graph, the meta-fields `input and output edges`, `input multiplicities` and `output multiplicities` are used to establish how many edges can go in and go out to and from a node. In the case of meta-table 3, which describes the concept dictionary, the meta-field `format` restricts the possibilities to fill in the cells (text, list, logic expression, etc). `Is it main` is true when the described field is the identifier of the row. `Repetition in the same table` is true when the field can be filled in with the same value in different rows. And `multiplicity` is true when the same cell can have several values.

[2] 'Subclass in a disjoint partition'. A disjoint partition of a class is a set of subclasses of this class that do not have common instances.

[3] 'Subclass in an exhaustive partition'. An exhaustive partition of a class is a set of subclasses that covers all the class, that is, there is not an instance of the father class that is not an instance of any of the subclasses of the partition.

Edge	Symbol	Description
Subclass of	S	A class C is a subclass of the parent class P if and only if every instance of C is also an instance of P.
Subclass in a disjoint partition	SDP	A disjoint partition of a class is a set of its subclasses where the subclasses do not have common instances.
Subclase in an exhaustive partition	SEP	An exhaustive partition of a class is a set of subclasses that covers all the class, that is, there is no instance of the father subclass that is not subclass of any class of the subclasses of the partition

Meta-table 1. `Meta-table of edge description` *defining the possible edges of the graph "concept classification tree"*

Node	Symbol	Descrip-tion	Input and outpud edges	Input multipli-cities	Output multipli-cities
Concept	C	**	Subclass of	(0, n)	(0, n)
			Subclass in a disjoint partition	(0, n)	(0, n)
			Subclass in an exhaus-tive parti-tion	(0, n)	(0, n)

Meta-table 2. `Meta-table of node description` *defining the possible nodes of the graph "concept classification tree"*

The meta-graph to model the order during the conceptualisation is not presented due to the space constraints. Concerning the consistency verification rules between tables, between graphs, and between tables and graphs, the way to write them is based on operations on matrices representing the tables and the graphs. Such operations are similar to the ones used in the relational model for databases (projection, selection, difference, etc.).

2.1.3. Meta-formalisation and meta-implementation of the conceptualisation process

To carry out the meta-formalisation, a formal and declarative language, called **LBIR** (Language for Building Intermediate Representations), has been elaborated. Such language has the same expressiveness as the meta-tables and meta-graphs used during the meta-conceptualisation. The LBIR description uses a context free grammar for the syntax, and matrices to establish the meaning of the language. The following code:

```
define table horizontal [Concept dictionary] as CD

define field  [Concept name]  as CN
          begin
            type term;
            repeated no ;
            multiplicity (1,1);
          end field  ;
define field  Instances as I
          begin
            type term;
            repeated yes;
            multiplicity (0,N);
define field  [Instance attributes] as IA
          begin
            type term;
            repeated yes;
            multiplicity (0,N);
          end field  ;
define field  Relations as R
          begin
            type term;
            repeated yes;
            multiplicity (0,N);
          end field  ;
begin
          placed in [Binary relation diagram];
          main field [Concept name];
end table ;
```

shows the definition in LBIR of the concept dictionary, that is equivalent to the definition appearing in meta-table 3. `Placed in binary relation diagram` indicates that a graph called binary relation diagram should be designed before filling in he concept dictionary.

Field	Symbol	Description	Format	Is it main?	Repetition in the same table	Multiplicity
Concept name	CN	**	Term	Yes	No	(1, 1)
Instances	I	Instances are particular cases of the concept	Term	No	Yes	(0, n)
Instance attributes	IA	The ones that allow describing the instances of the concept.	Term	No	Yes	(0, n)
Relations	R	Relations link concepts	Term	No	Yes	(0, n)

Meta-table 3. `Meta-table of field description` *defining the table "concept dictionary"*

During the meta-implementation, the meta-model expressed in LBIR is transformed into a SQL schema. This eases the use of *databases to store ontologies*, taking advantage of the independence and integrity of the data, the minimisation of the redundancy, etc., provided by the relational database systems.

3.2. THE SOFTWARE LEVEL OF THE BI-PHASE SOLUTION

In order to allow the efficient use of the methodology proposed in 3.1, we has built ODE (see figure 5). The *LBIR processing module* automates the transformation, without loss of expressiveness, from a meta-model in LBIR to a SQL schema. Besides, it allows conceptualising ontologies following a meta-model selected by the user, and storing the result in a database following the SQL schema associated to the meta-model. Moreover, if you follow the reference meta-model to conceptualise your ontology you can use a generator of Ontolingua code. The main feature of ODE is that a change in the meta-model does not force a change in the program, since SQL schemas are generated in run-time and not in design time (as usual).

3. CONCLUSIONS AND FUTURE TRENDS

Although each ontology has its modelling needs, there is not any methodological proposal to use a different kind of modelling for each ontology.

The bi-phase method presented in this paper proposes, during a first phase, to model the modelling process itself (or reusing an existing meta-model) and, during the second phase, to model the ontology. In the first phase, the steps are: meta-specification, meta-conceptualisation, meta-formalisation and meta-implementation. During the second phase the steps are the ones proposed by METHONTOLOGY: specification, conceptualisation, formalisation and implementation. To carry out the meta-formalisation, a formal and **declarative language** (LBIR) has been elaborated. Moreover, to provide software support to both phases, a tool has been developed: ODE.

To agree in the meta-model to be used, the different groups can exchange this meta-model in meta-tables and meta-graphs, or in LBIR (see figure 6). The second option, LBIR, is mandatory if the current version of ODE is utilised to build the ontologies.

The method and the tool have proved useful in several Spanish and international projects.

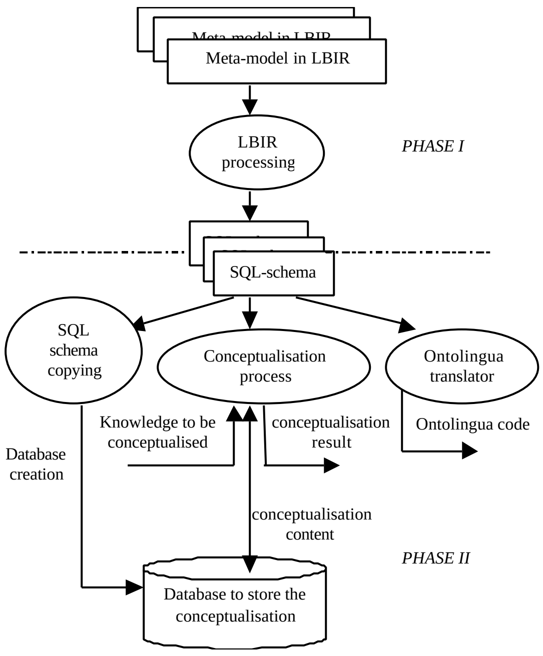


Figure 5. ODE processes

One of the most interesting future lines, above all for ODE, would be the fast development of translators from different meta-models into different implementation languages. An interface to manipulate meta-tables and meta-grpahs would be also interesting. Another important future trend would be a structured characterisation of ontologies according to their modelling needs. Now, the modelling necessities are determined by the experience of the ontology engineers, who interacts with the experts in the domain to be modelled.

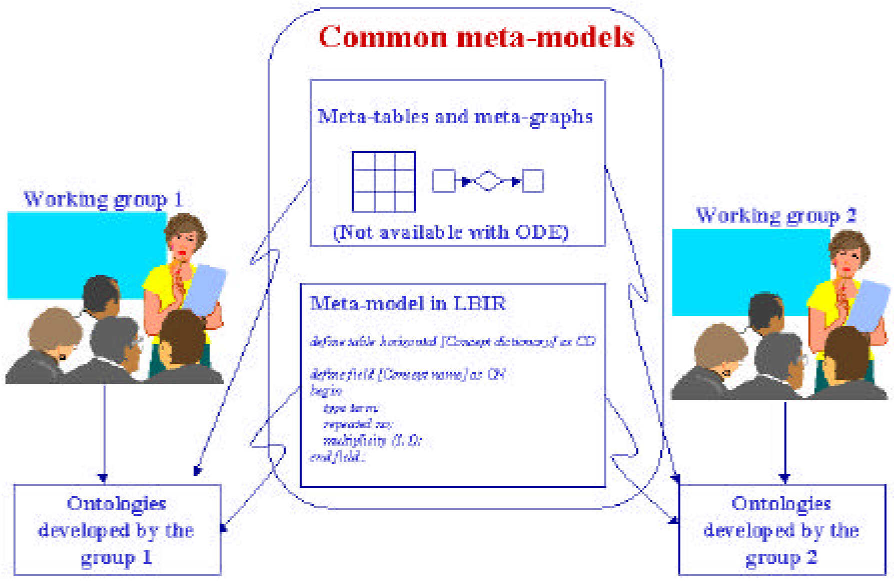

Common meta-models
Meta-tables and meta-graphs
(Not available with ODE)
Meta-model in LBIR
define table horizontal [Concept dictionary] as CD
define field [Concept name] as CN
begin
type term;
repeated no;
multiplicity (1, 1);
end field;
Working group 1
Working group 2
Ontologies developed by the group 1
Ontologies developed by the group 2

Figure 6. Use of meta-models for cooperative construction of ontologies

4. REFERENCES

[Ben98] Benjamins, V.R.; Fensel, D.; Gómez-Pérez, A.; Decker, S.; Erdmann, M.; Motta, E.; Musen, M. 1998. "The Knowledge Annotation Initiative of the Knowledge Acquisition Community $(KA)^2$". In: *Proceedings of the 11th Banff Knowledge Acquisition for Knowledge-Based System Workshop (KAW 98)*, Banff, Canada.

[Eri99] Eriksson, H.; Fergerson, R.W.; Shahar, Y.; Musen, M.A. "Automatic generation of ontology editors". Knowledge Acquisition Workshop (KAW). Banff (Canada). 1999.

[Far97] Farquhar, A.; Fikes, R.; Rice, J. "The Ontolingua Server: Tool for Collaborative Ontology Construction". IJHCS, 46(6) (1997) 707-728.

[Fen00] Fensel, D. *Ontologies: silver bullet for Knowledge Management and Electronic Commerce*. Springer-Verlag. Berlin. 2000.

[Fer01] Fernández-López, M. *Método bi-fase para la conceptualización de ontologías basado en meta-modelos*. Ph. Thesis. Facultad de Informática. Universidad Politécnica de Madrid. Spain. 2001.

[FeG99] Fernández-López, M.; Gómez-Pérez, A.; Pazos-Sierra, A.; Pazos-Sierra, J. "Building a Chemical Ontology Using METHONTOLOGY and the Ontology Design Environment". *IEEE Intelligent Systems & their applications*. January/February. Pages 37-46. 1999.

[Fer99] Fernández López, M. "Overview of Methodologies for Building Ontologies". *Workshop on Ontologies and Problem-Solving Methods: Lessons Learned and Future Trends. (IJCAI99)*. August. 1999.

[Fri00] Fridman Noy, N.; Fergeson, R.W.; Musen, M.A. "The knowledge model of Protégé-2000: combining interoperability and flexibility". *Workshop on Knowledge Acquisition, Modelling and Management (EKAW)*. Editor Springer Verlag. Jean Les Pins (Francia). 2000.

[Grü95] Grüninger, M.; Fox, M. S. "Methodology for the design and evaluation of ontologies". *Workshop on Basic Ontological Issues in Knowledge Sharing. Montreal*, Canada. 1995.

[Hay83] Hayes-Roth, F.; Waterman, D. A.; Lenat, D. B. "Building Expert Systems". *Addison Wesley Publishing Company, Inc*. Massachusets, USA. 1983.

[Nec91] Neches, N.; Fikes, R.; Finin, T.; Gruber, T.; Patil, R.; Senator, T.;

Swartout, W.R. “Enabling technology for knowledge sharing”. *AI Magazine, Fall* 1991. Pages 36-56. 1991.

[New82] Newell, A. “The Knowledge Level”. Artificial Intelligence. Volume 18. Number 1. Pp. 87-127. January 1982.

[Usc95] Uschold, M. King, M. “Towards a methodology for building ontologies”. *Workshop on Basic Ontological Issues in Knowledge Sharing*. *International Joint Conference on Artificial Intelligence* (IJCAI). Montreal, Canada. 1995.